CHARUSAT
CHAROTAR UNIVERSITY OF SCIENCE AND TECHNOLOGY

CE144:OBJECT ORIENTED PROGRAMMING WITH C++

February 2023 – May 2023

UNIT 2

INTRODUCTION TO C++

DEPSTAR **Devang Patel Institute of Advance Technology and Research**

Objectives

- Learning the basics of C++ and its applications.
- Creating a simple C++ Program.

Introduction

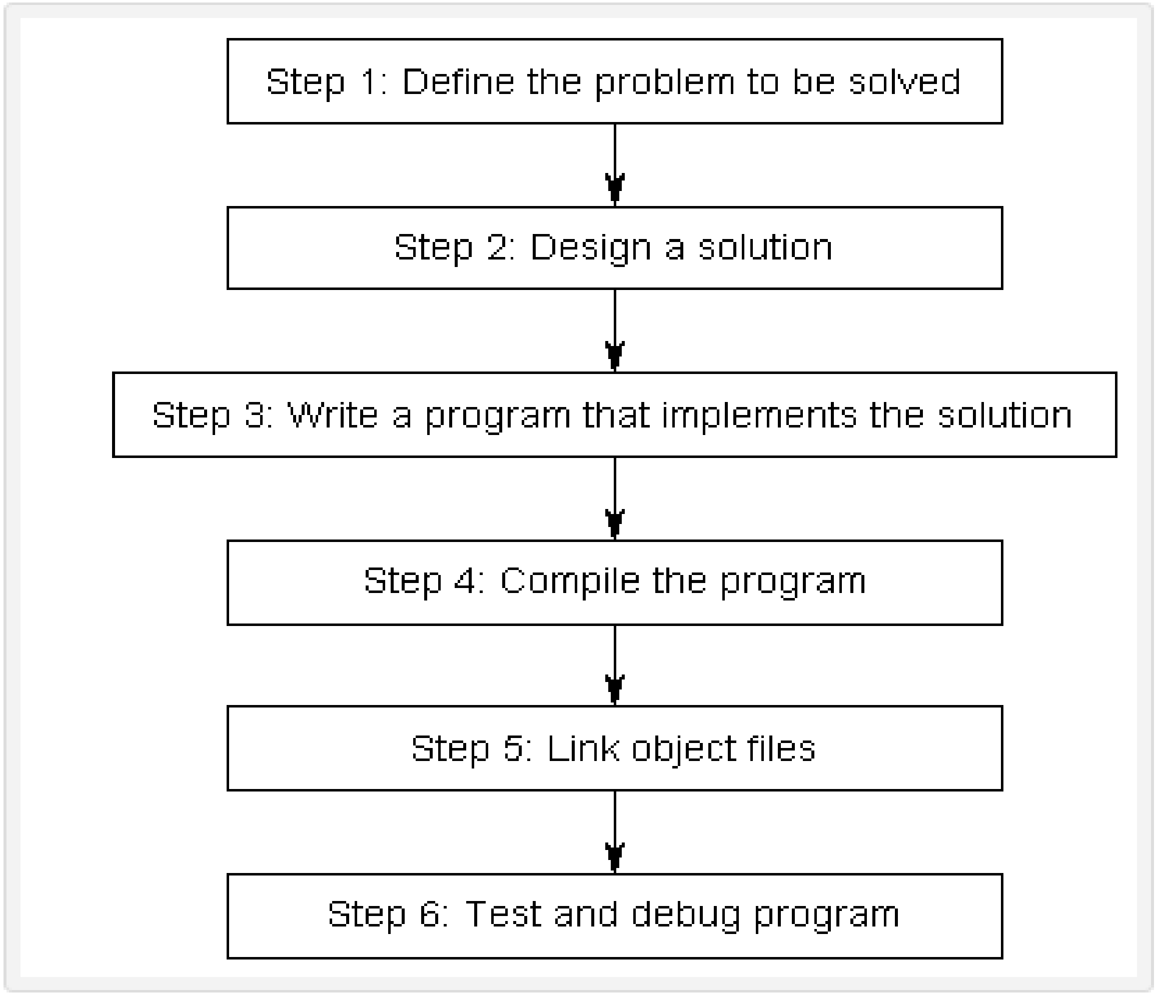
In this chapter, we will discuss

- What is C++?
- Simple C++ Program
- Applications of C++
- Introduction to class, object and creating a simple C++ program
- Structure of C++ Program

C++

- C++ Developed in 1979
  - Developed by Bjarne Stroustrup
  - Developed at Bell Telephone laboratories
  - Originally called “C with classes”
  - The name C++ is based on C’s increment operator (++)
    - Indicating that C++ is an enhanced version of C

Development of Program

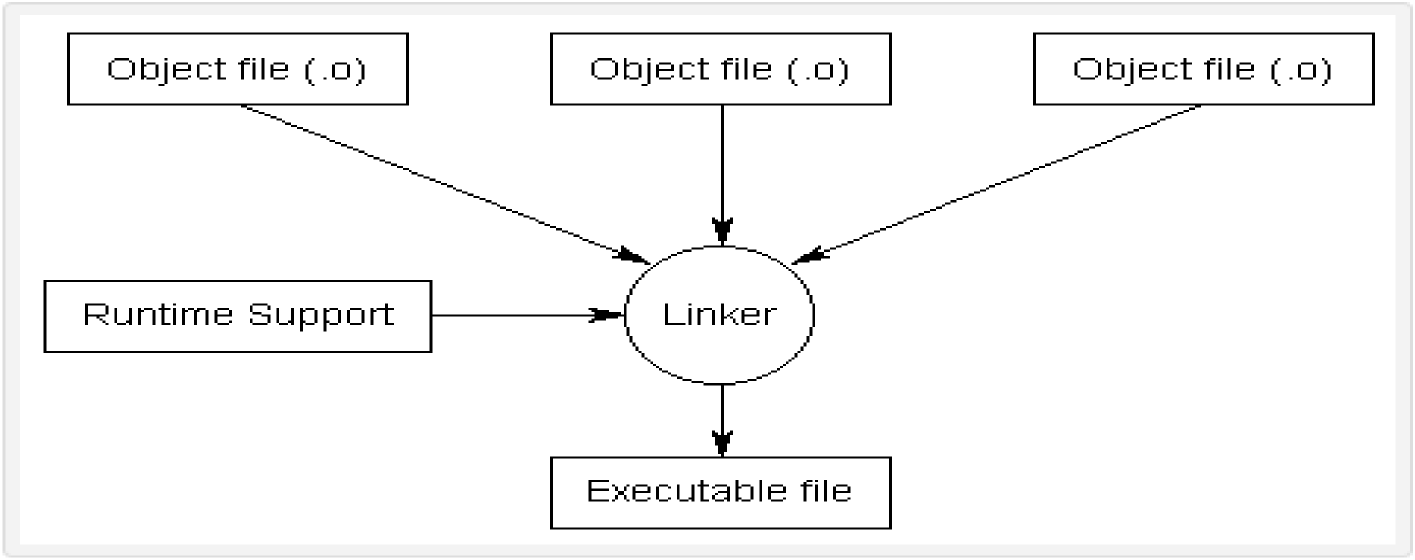

Step 1: Define the problem to be solved
Step 2: Design a solution
Step 3: Write a program that implements the solution
Step 4: Compile the program
Step 5: Link object files
Step 6: Test and debug program

Compiling

- The job of the compiler is:
  - To check your program and make sure it follows the rules of the C++ language
    - If it does not, the compiler will give you an error to help pinpoint what needs fixing
  - To convert each file of source code into a machine language file called an **object file**

Linking

- Process of taking all the object files generated by the compiler and combining them into a single executable program that you can run
- This is done by a program called the **linker**



A Simple C++ program

The iostream file

```
#include <iostream>     // include header file
using namespace std;
int main()
{
   cout<<"C++ is better than C. \n";
   return 0;
}
```

Comments

Namespace (std where standard class libraries defined)

Output Operator

A C++ program

```
/* include headers; these are modules that include functions that you may use in
   your program; we will almost always need to include the header that defines cin
   and cout; the header is called iostream.h */

#include <iostream.h>

int main() {
//variable declaration
//read values input from user
//computation  and print output to user
return 0;
}
```

After you write a C++ program you compile it; that is, you run a program called **compiler** that checks whether the program follows the C++ syntax

- if it finds errors, it lists them
- If there are no errors, it translates the C++ program into a program in machine language which you can execute

Contd..

- what follows after // on the same line is considered comment
- indentation is for the convenience of the reader; compiler ignores all spaces and new line ; the delimiter for the compiler is the semicolon
- all statements ended by semicolon
- **Lower vs. upper case matters!!**
  - Void is different than void
  - Main is different that main

Hello world program

Here is the Hello world program in C++.

```
#include <iostream.h>

int main()

{

cout << “Hello world!”;

return 0;

}
```

Contd..

```
// my first program in C++
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

Contd..

- **#include <iostream>**

➢ Lines beginning with a hash sign (#) are directives read and interpreted by what is known as the *preprocessor*. They are special lines interpreted before the compilation of the program itself begins.

➢ In this case, the directive #include <iostream>, instructs the preprocessor to include a section of standard C++ code, known as *header iostream*, that allows to perform standard input and output operations, such as writing the output of this program (Hello World) to the screen.

Contd..

- **std::cout << "Hello World!";**

- This line is a C++ statement.
- A statement is an expression that can actually produce some effect.
- Statements are executed in the same order that they appear within a function's body.

Using namespace std

- If you have seen C++ code before, you may have seen cout being used instead of std::cout.
- Both name the same object: the first one uses its unqualified name (cout), while the second qualifies it directly within the namespace std (as std::cout).
- cout is part of the standard library, and all the elements in the standard C++ library are declared within what is called a namespace: the namespace std.

Contd..

- In order to refer to the elements in the std namespace a program shall either qualify each and every use of elements of the library (as we have done by prefixing cout with std::), or introduce visibility of its components. The most typical way to introduce visibility of these components is by means of using declarations:
    - **using namespace std;**

Example

```
// my second program in C++
#include <iostream>
using namespace std;
int main ()
{
cout << "Hello World! ";
cout << "I'm a C++ program";
}
```

more C++ statements

```
#include <iostream>

using namespace std;

int main()
{
    float number1, number2, sum, average;
    cout    <<  "enter two numbers: ";
    cin >>  number1;
    cin >> number2;

    sum =   number1 +   number2;
    average =   sum/2;

    cout    <<  "sum    ="  <<  sum <<  "\n";
    cout    <<  "Average    ="  <<  average <<  "\n";

    return  0;
}
```

Variable

Input Operator

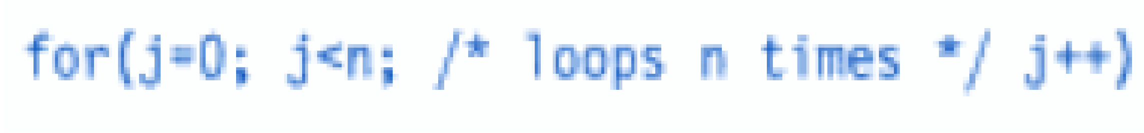
Cascading of I/O Operators

Comments

```
// This is a C++ program. It prints the sentence:
// Welcome to C++ Programming.
```

Or

```
/*
    You can include comments that can
    occupy several lines.
*/
```



```
for(j=0; j<n; /* loops n times */ j++)
```

✗

Variable declaration

type variable-name;

Meaning: variable <variable-name> will be a variable of type <type>

Where type can be:

– int	//integer
– double	//real number
– char	//character

Example:

```
int a, b, c;
double x;
int sum;
char my-character;
```

Input statements

cin >> variable-name;

Meaning: read the value of the variable called <variable-name> from the user

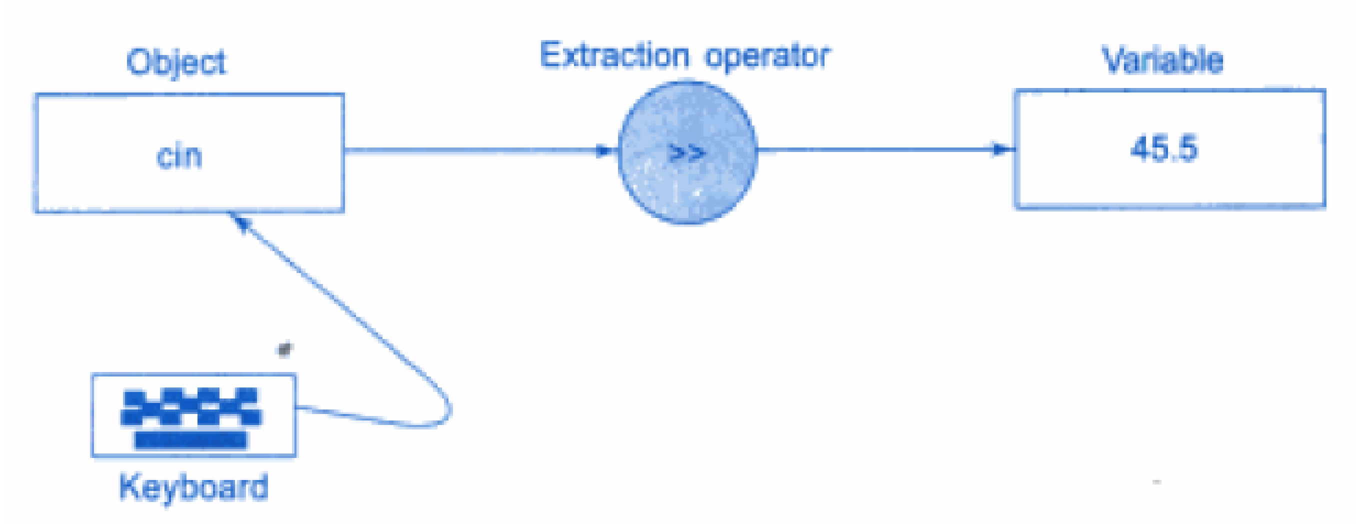
- The operator >> is known as the **extraction or get from** operator.
- It extracts the value from the keyboard and assigns it to the variable on the right.

Example:

```
cin >> a;
cin >> b >> c;
cin >> x;
cin >> my-character;
```

Input Operator

```
cin >> number1;
```



The operator >> is known as *extraction or get from* operator.

Output statements

cout << variable-name;

Meaning: print the value of variable <variable-name> to the user

cout << “any message”;

Meaning: print the message within quotes to the user

cout << endl;

Meaning: print a new line

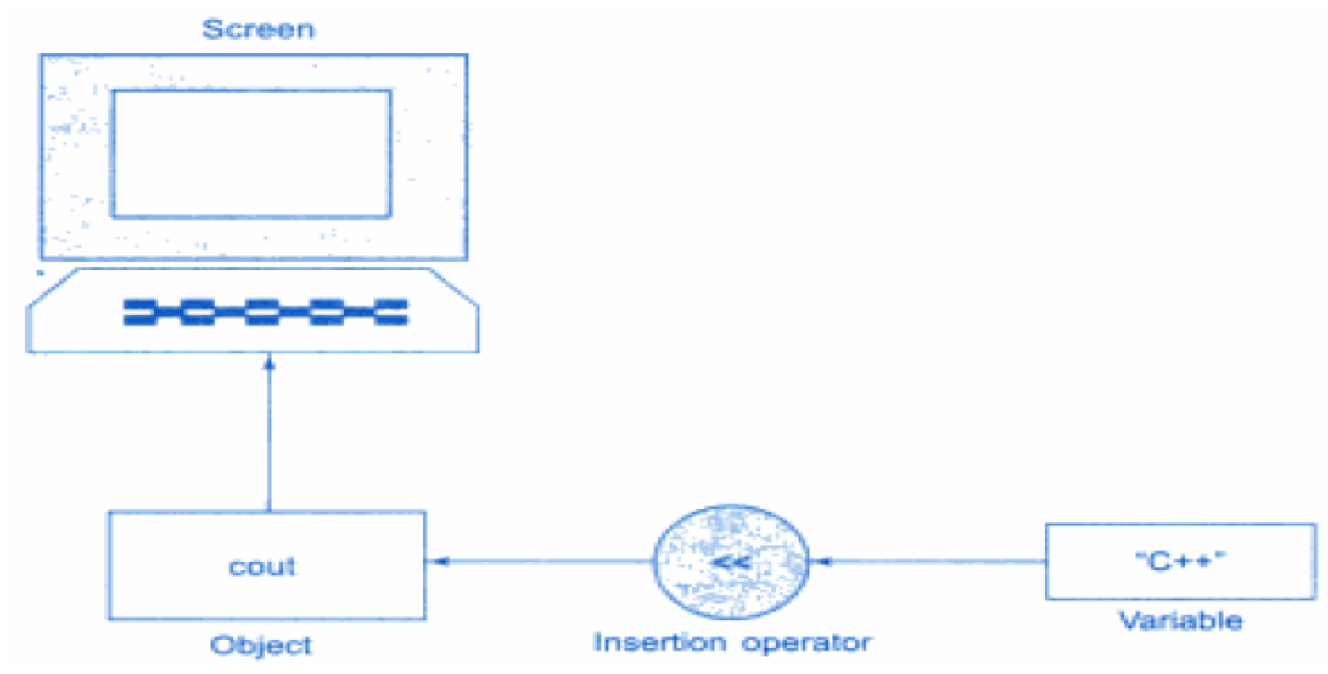
- The operator << is known as the **insertion operator**.
- Multiple use of << in one statement is known as **cascading.**

Example:

```
cout << a;
cout << b << c;
cout << “This is my character: “ << my-character << “ he he he”
        << endl;
```

Output Operator

```
cout<< "c++ is better than c";
cout<<name;
```



Output using insertion operator

The operator << is called as insertion or put to operator

Cascading I/O Operators

- The multiple use of << or >> in one statement is called cascading.
- Example:-

  cout<< “ sum”<< sum << “\n”

- First sends the string “sum=“ to cout and then sends the value of sum.
- Finally, it sends the newline character so that the next output will be in the new line.

Example:-

cin>>number1>>number2;

Access Modifiers

- Accessing a data member depends solely on the access control of that data member. This access control is given by Access modifiers in C++.
- There are three access modifiers :

1. **public**
2. **private**
3. **protected**

Contd..

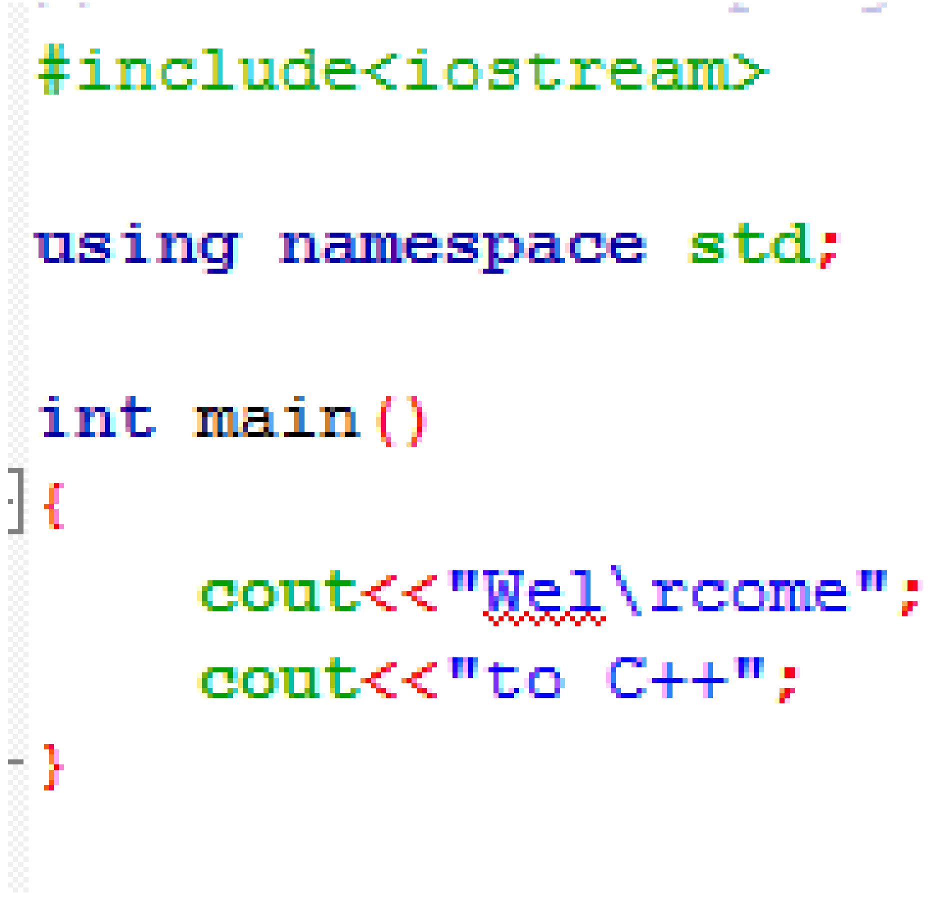
- **Public:** A **public** member is accessible from anywhere outside the class but within a program.
- **Private:** A **private** member variable or function cannot be accessed, or even viewed from outside the class. Private members **can only be accessed by member functions of the same class or friends.** This means derived classes can not access private members of the base class directly!
- **Protected:** A **protected** member variable or function is very similar to a private member but it provided one additional benefit that they can be accessed in child classes which are called derived classes.

Creating Simple program using class

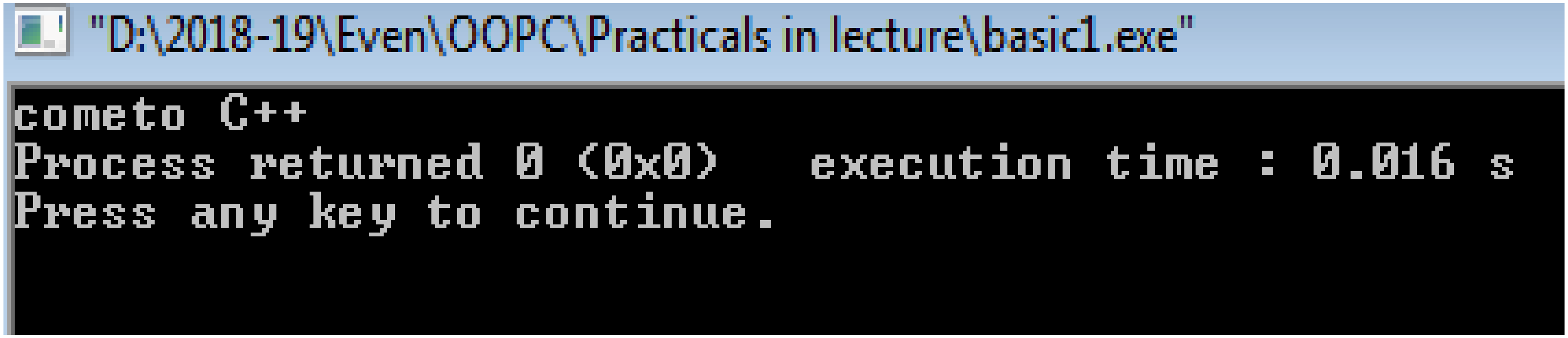
```
#include<iostream>
using namespace std;
void show();
class Test
{
public:
    void show()
    {
        cout << "I am a member function of class"<<"\n";
    }
};
int main()
{
Test t;
t.show();
show();
}
void show()
{ cout << "I am a function defined outside the class"<<"\n";}
```

Common escape sequences

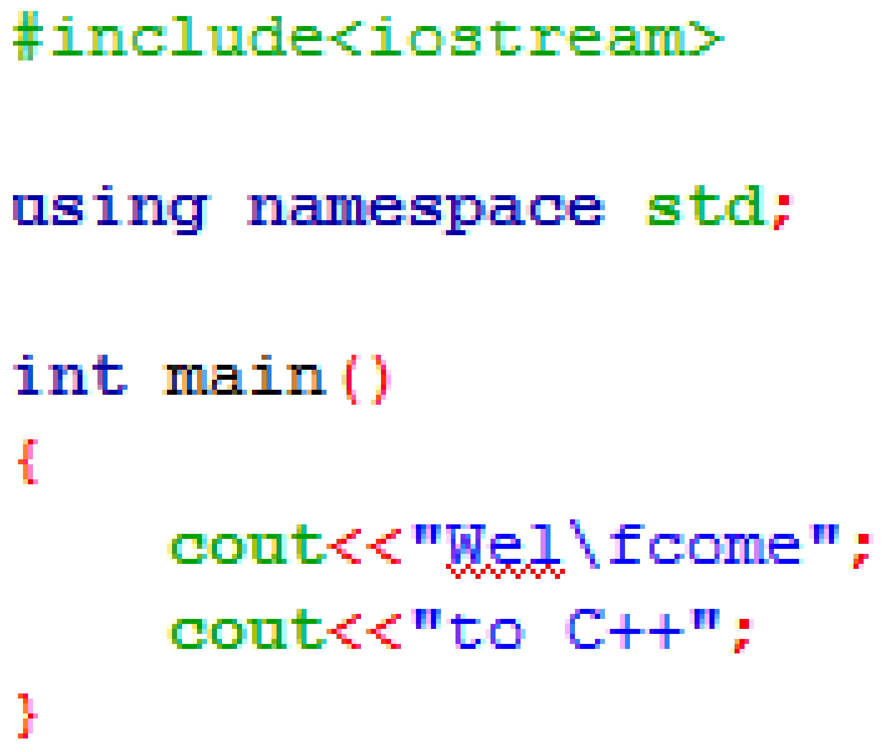
Escape Sequences	Character
\a	Bell (beep)
\b	Backspace
\f	Formfeed
\n	Newline
\r	Return
\t	Tab
\\	Backslash
\’	Single quotation mark
\”	Double quotation marks
\xadd	Hexadecimal representation



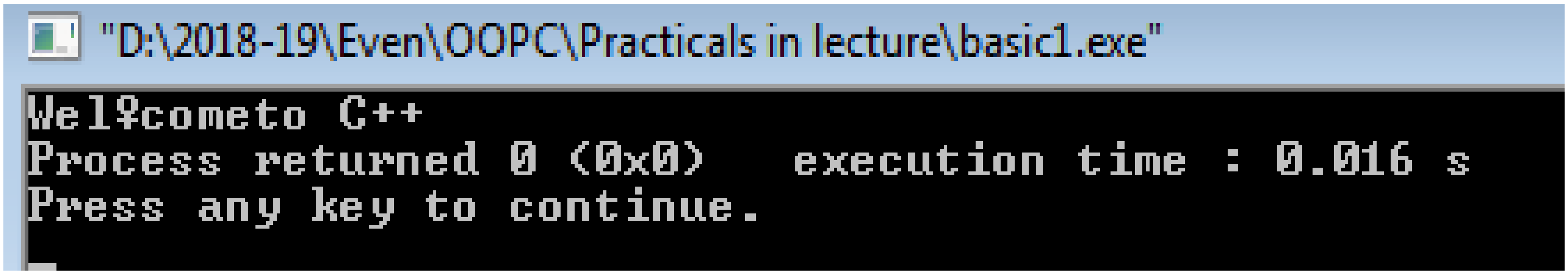
```
#include<iostream>

using namespace std;

int main()
{
    cout<<"Wel\rcome";
    cout<<"to C++";
}
```



```
"D:\2018-19\Even\OOPC\Practicals in lecture\basic1.exe"
cometo C++
Process returned 0 (0x0)   execution time : 0.016 s
Press any key to continue.
```

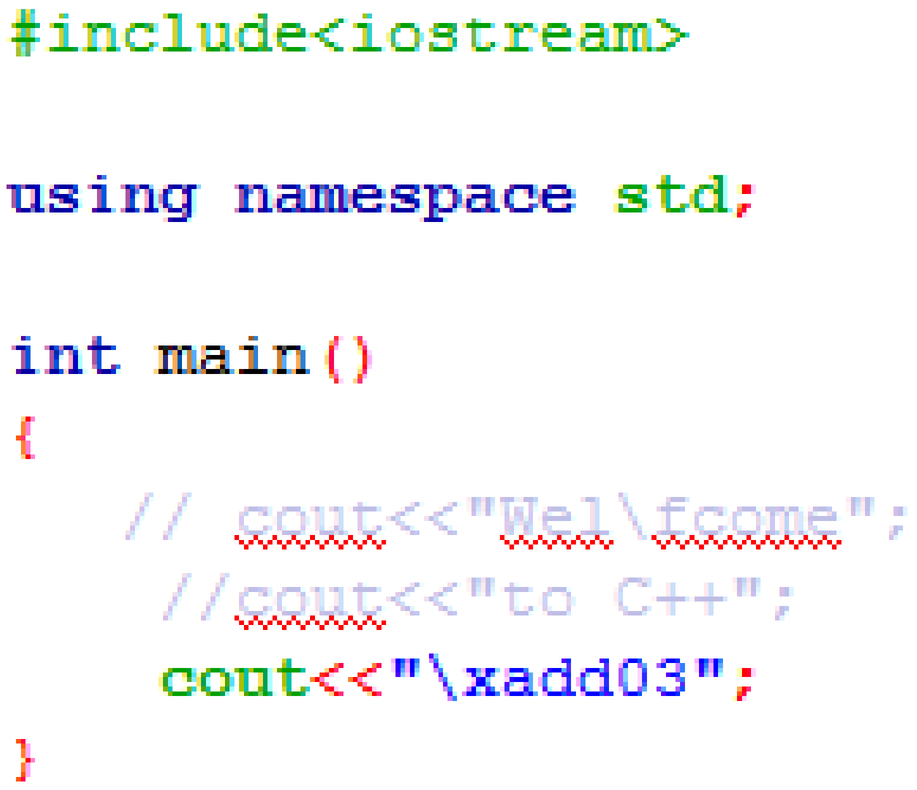


```cpp
#include<iostream>

using namespace std;

int main()
{
	cout<<"Wel\fcome";
	cout<<"to C++";
}
```

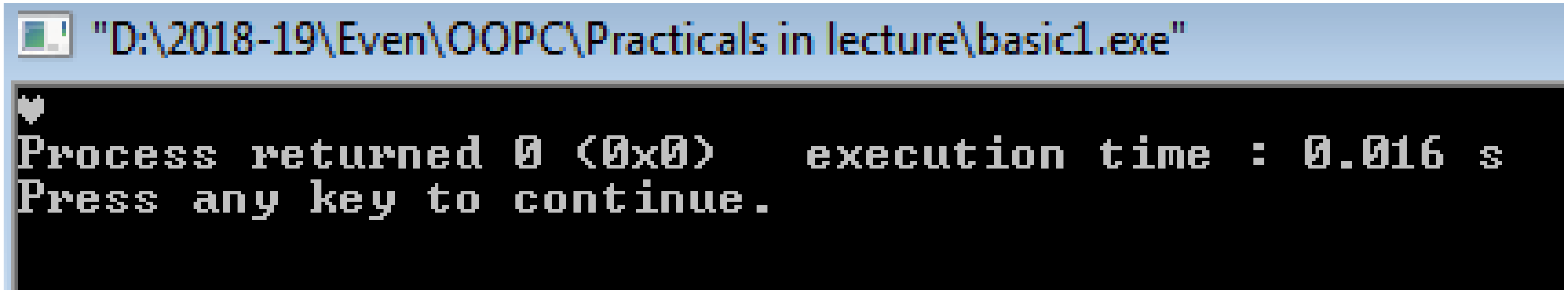


"D:\2018-19\Even\OOPC\Practicals in lecture\basic1.exe"

```
Wel♀cometo C++
Process returned 0 (0x0)   execution time : 0.016 s
Press any key to continue.
```



```
#include<iostream>

using namespace std;

int main()
{
    // cout<<"Wel\fcome";
     //cout<<"to C++";
     cout<<"\xadd03";
}
```



"D:\2018-19\Even\OOPC\Practicals in lecture\basic1.exe"

```
♥
Process returned 0 (0x0)   execution time : 0.016 s
Press any key to continue.
```

Structure of C++ program

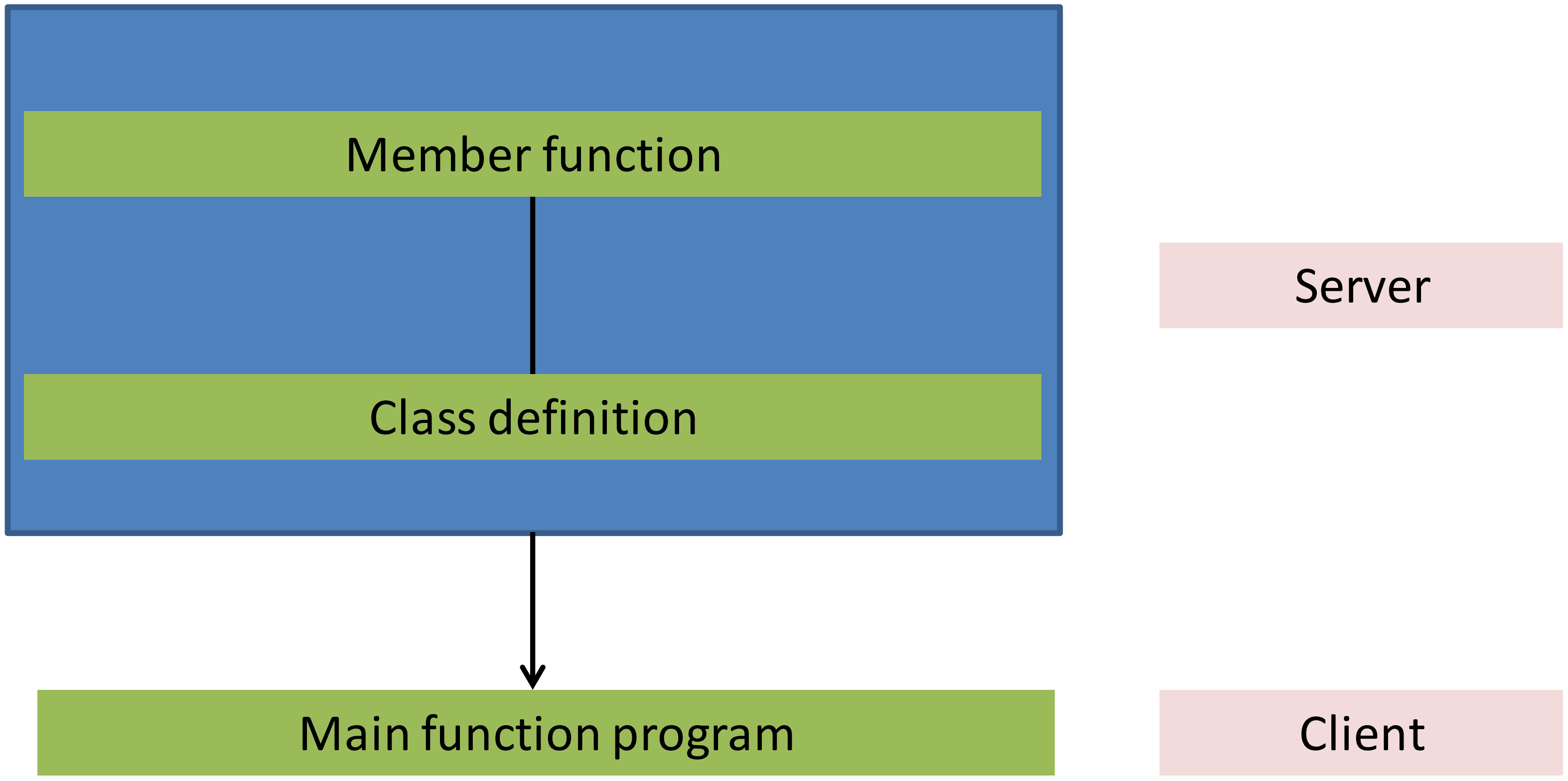
Include files

Class declarations

Member functions definitions

Main function program

The client-server model



Real World Applications of C++

- Games
- Graphic User Interface (GUI) based applications
- Web Browsers
- Advance Computations and Graphics
- Database Software
- Operating Systems
- Enterprise Software
- Medical and Engineering Applications
- Compilers

Versions of C++

1. Apple C++
2. Borland C++
3. GNU C++ for Linux
4. IBM C++ for IBM power, System Z
5. SGI C++
6. Sun C++
7. Microsoft Visual C++ under .Net
8. Turbo C++

C++ versus C

The following features of C++ language or library is not supported in C:

1. Classes
2. Constructors and destructors
3. Exceptions and try/catch block
4. Function overloading
5. Member functions
6. Namespaces
7. New and delete operators
8. Operator overloading
9. Standard template library (STL) [cmath, iostream, cstdlib, cstring and many more]

Possible Future Additions to C++

It is speculated that the following new features will be added to C++ in future:

1. Automatic garbage collection
2. Object persistence
3. Support for concurrency and multithreading
4. Extensible member functions
5. Dynamically linked libraries
6. Rule based programming

Summary

- C++ supports interactive input and output features.
- A typical C++ program would contain 4 basic sections namely file section, class declaration section, member function section and main function section.

Thank You